Variable Resolution 4–k Meshes

LUIZ VELHO JONAS GOMES

IMPA–Instituto de Matemática Pura e Aplicada,
{lvelho|jonas}@visgraf.impa.br

Abstract. In this paper we introduce variable resoultion 4–k meshes, a powerful structure for the representation of geometric objects at multiple levels of detail. It combines most properties of other related descriptions with several advantages, such as more flexibility and greater expressive power.

The main unique feature of the 4–k mesh structure lies in its variable resolution capability, which is crucial for adaptive computation. We also give an overview of the different methods for constructing the hierarchical 4–k mesh representation, as well as the basic algorithms necessary to incorporate it in modeling and graphics applications.

Keywords: multiresolution, four-directional grids, restricted quad-trees, multi-triangulations, adapted meshes.

1 Introduction

Hierarchical structures are the embodiment of fundamental abstraction mechanisms that allow us to deal with complexity. For this reason, such structures are an integral part of many tools in practically every area of human activity.

Hierarchies reflect dependency relations between entities at different levels. The specific nature of these relationships is determined by the application area, and by the problem to be solved.

In Geometric Modeling and Computer Graphics, hierarchical structures are often used to represent objects with multiple levels of detail. This type of hierarchy makes it possible to process the object at different resolutions. Thus, hierarchical structures are essential for most algorithms that require adapted computations. A typical example is the visualization of 3D polygonal surfaces, where the size of polygonal facets should be proportional to the projected area on the screen.

The importance of multiple levels of detail representations has motivated the development of hierarchical structures which, in one way or another, support that capability.

In this paper, we present the hierarchical 4–k mesh structure. It combines most properties of other multiple level of detail representations and offers several advantages.

2 Basic Concepts

This section gives some definitions and basic notions that are used throughout the paper.

A mesh is a cell complex, $K = (V, E, F)$, where V, E and F are respectively sets of vertices $v_i \in V$, edges $(v_i, v_j) \in E$, and faces $(v_i, v_j, \ldots, v_k) \in F$. The complex K provides a *topological structure* for the decomposition of two dimensional domains.

A *geometric realization* of the mesh K is created, by associating to each vertex $v_i \in V$, a coordinate value, $p(v_i) \in \mathbb{R}^n$. When $n = 2$, K is a planar mesh and when $n = 3$, K is a surface in 3D.

A mesh is called *conforming* when faces that are spatially adjacent share exactly edges and vertices on common boundaries.

The 1-neighborhood $N_1(v)$ of a vertex v consists of the set of vertices that share a face with v. The *valence* (or *degree*) of a vertex, v, is the number of edges incident in v.

The mesh structure is related to the types of 1-neighborhoods in the mesh. In a *regular mesh*, the valence of all vertices is the same, while in an *irregular mesh* the valence may differ from vertex to vertex in an arbitrary way.

The size of a mesh, denoted by $|K|$, is the number of faces in the set F of K.

The *resolution* of a uniform mesh is the number of vertices per unit length. The resolution of an irregular mesh can be determined locally from the length of its edges.

Two meshes K_m and K_n are *compatible*, if there is a subset of faces $\overline{F}_m \in K_m$, that when the corresponding subset of faces in K_n is replaced by $\overline{F}_m$ the result is a conforming mesh. Correspondence in this case means spatial overlap.

A *mesh hierarchy*, H, is a sequence of meshes, $H = (K_j)_{j=1,\ldots,n-1}$, such that the size of the mesh K_j increases monotonically with the index j. Furthermore, there is a dependency relation between faces at two subsequent levels j and $j+1$, whose supports overlap.

Based on these dependency relations, it is possible to construct a hierarchical structure that defines the increasing sequence of meshes H (it is also possible to define the *reverse* of the hierarchy, which is the sequence in reverse order, where the mesh size is decreasing).

A mesh hierarchy is usually constructed by local modifications that either refine or simplify some initial mesh. Thus, one can start with a coarse mesh and subdivide it by applying a refinement operator; or, alternatively, one can

0-7695-0878-2/00 $10.00 © 2000 IEEE

start with a fine mesh and coarsify it by applying a simplification operator. Figure 1 shows a scheme of this process.

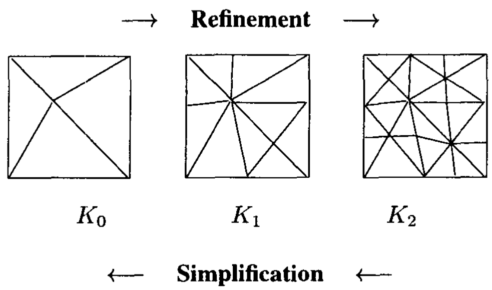


Figure 1: Mesh hierarchy and construction mechanisms.

Note that the modification operator provides the dependency relations necessary to build a hierarchical structure encoding the mesh hierarchy.

The nature of these operators and the method by which they are applied determines the properties of the hierarchy.

Here, we distinguish between hierarchical structures of two kinds: *non-adaptive* and *adaptive*.

Non-adaptive hierarchical structure: defines only one mesh hierarchy. Examples of this kind of structure are multiresolution and progressive meshes.

In a *multiresolution mesh*, the modifications are applied in parallel to a set of independent regions that completely cover the mesh. Each step of this process changes the mesh resolution globally. The corresponding hierarchical data structure is a tree. A multiresolution structure is usually constructed using refinement [10].

In a *progressive mesh*, the modifications are applied sequentially to only one region of the mesh at time. Each step of the process changes the mesh resolution locally. The corresponding data structure is a list. The progressive structure is usually constructed using simplification [5].

Adaptive hierarchical structure: defines a family of mesh hierarchies. One example of this kind of structure is a variable resolution mesh.

In a *variable resolution mesh*, local modifications are applied to a set of independent regions, in such a way that the boundary of each region remains unchanged. Note that this set of regions may not cover the mesh completely. Because of the boundary constraint, there is no interference between local modifications at each level, which can be applied independently of each other. The above property makes it possible to generate many sequences of meshes using permutations of independent local modifications. The corresponding data structure is a directed acyclic graph (DAG), that encodes dependencies across levels. A variable resolution mesh can be constructed either by refinement or simplification [9].

3 Variable Resolution Triangulations

This section defines more precisely some basic notions concerning adaptive hierarchical structures.

The idea of a variable resolution triangulation was introduced independently by Puppo [8] and De Berg et al. [2]. Subsequently, Puppo developed an extensive theoretical framework for general variable resolution structures, which he called *Multi-Triangulations*. This section summarizes these concepts.

As mentioned above, hierarchical mesh structures are based on local modifications. In the variable resolution setting, it is necessary to employ a restricted class of local modifications: the ones that are minimally compatible.

A *minimally compatible local modification*, $W(K_i)$, to a sub-mesh $K_i \subset K$ of a mesh $K = (V, E, F)$, is a substitution of K_i by $W(K_i)$ in K, such that:

1. The boundary edges of K_i are not altered (except when they also belong to the boundary of the mesh K);
2. The interior edges of K_i are replaced by new edges.

The sub-meshes K_i and $W(K_i)$ are, respectively, the *pre-image* and *image* of the modification operator W.

Compatibility is enforced by condition (1). Since the boundary ∂K_i does not change, the new sub-mesh $W(K_i)$ is compatible with K_i and the modification operator produces a conforming mesh.

Minimality is addressed by condition (2). Since the interior of K_i changes completely, there is minimal redundancy between the sub-meshes K_i and $W(K_i)$.

The modification operator W is called *increasing* if $|W(K_i)| > |K_i|$. This means that W is a refinement operator. Similarly, W is called *decreasing* if $|W(K_i)| < |K_i|$. In this case, it is a simplification operator.

A *compatible sequence of meshes*, $(K_0, K_1, \ldots, K_n)$, is generated by the application of a sequence of modifications $(W_1, W_2, \ldots, W_{n-1})$, starting with an initial mesh K_0. This produces the sequence of meshes $(K_0, W_1(K_1), \ldots, W_{n-1}(K_{n-1}))$, where

$$K_j = W_{j-1}(W_{j-2}(\cdots W_1(K_1))) \qquad (1)$$

for $j > 0$.

Note that, given an intermediate mesh, K_m, and two independent modifications W_j and W_l, that are compatible with K_m, we can apply either one of them to K_m, in order to produce a new mesh $K_{m+1} = W_j(K_j)$ or $K_{m+1} = W_l(K_l)$, with $K_j, K_l \subset K_m$.

The purpose of a variable resolution structure is to encode **all** possible mesh hierarchies that can be generated from a sequence of modifications $(W_i)_{i=1,\ldots,n-1}$. In order to achieve this goal, we need to distinguish between dependent and independent modifications.

A *variable resolution mesh*, $V = (K_0, W, \leq)$ is defined by an initial mesh K_0, a set of minimally compatible local modifications $W = \{W_1, W_2, \ldots, W_{n-1}\}$, and a partial order relation $\leq$ on W, that satisfies the following conditions:

- *Dependency*: $W_i \prec W_j$, if and only if there is a face $f \in F_i$ in the pre-image K_i of W_i that belongs to the image $W_j(K_j)$ of W_j. In other words, precedence is determined by compatibility of dependent modifications.
- *Non-redundancy*: $f \in F_i$ of $W_i(K_i)$ implies that $f \notin F_j$ of $W_j(K_j)$ for all $j \neq i$. In other words, there are no duplicate faces.

The partial order relations can be described by a directed acyclic graph (DAG), where the nodes are associated with modifications W_i, and there is an arc from W_i to W_j whenever W_j is the successor of W_i according to the partial order relation $\leq$.

We construct a *lattice representation* of a variable resolution mesh by adding a source and a drain to the DAG.

In this representation, each face f is referenced by exactly two nodes. It appears in the image and in the pre-image of a modification. The node having f in its pre-image is called *successor* of f, and the node having f in its image is called *predecessor* of f.

The source node is associated with a constructor of the initial mesh K_0, and the drain node is associated with the application of all modifications W_i, $i = 1, \ldots, n-1$, to K_0, that produces the final mesh K_n. Appropriate arcs are added to and from these two special nodes.

A cut of a DAG consists of a set of nodes disconnecting it. A *front* in a lattice is a cut which contains exactly one arc for each path from the source to the drain.

Figure 6 shows an illustration of the representation of a variable resolution 4-*k* mesh and a cut in it.

4 Hierarchical 4–k Meshes

This section describes a hierarchical structure to encode the family of mesh hierarchies, which we call variable resolution 4–*k* meshes. This mesh is a special case of the variable resolution triangulation, defined in Section 3. Because of its particular nature, it has unique desirable properties not available in other general hierarchical structures.

A *variable resolution 4–k mesh* is a hierarchical structure that contains at each level approximately half of its vertices of valence 4 and other vertices of arbitrary valence *k*.

The 4–*k* mesh is built from a restricted set of local modifications defined on a cluster of two triangular faces. These two modifications are:

i. Internal edge split: the edge shared by two adjacent faces is subdivided, and the two faces are replaced by four faces. See Figure 2.

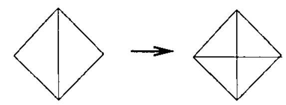


Figure 2: Internal edge split.

ii. Internal edge swap: The edge shared by two adjacent faces is replaced by another edge linking the opposite vertices in each face. See Figure 3

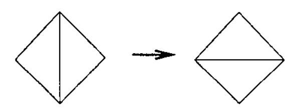


Figure 3: Internal edge swap.

Note that, for a planar mesh, these modifications make sense only if the two-face cluster is convex. Also note that modification (i) is a binary subdivision refinement applied to adjacent faces. The inverse of (i) is an edge collapse. It can be shown that these operations are sufficient to make any topology preserving transformation to a mesh [6].

Another important observation is that both (i) and (ii) are edge-based modifications. We exploit this fact to design data structures for representing variable resolution meshes.

The description combines edge and face elements. Modifications of type (i) are associated with an edge that splits or collapses. The edge points out to the two faces sharing it. Additionally, a face points out to its parent and two children.

This representation is illustrated in Figure 4.

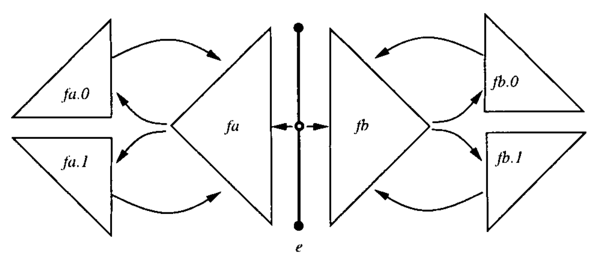


Figure 4: Edge-face 4–8 variable resolution structures

The specification of these data structures in pseudo-C is given below. A face is represented by the structure:

```
Face {
  Hedge*   edge[3];
  Face*    parent, children[2];
}
```

where we adopt the convention that the split edge of a face is `edge[0]`. (i.e. `split_edge(f):=f.e[0]`). When a face is bisected its edges are cyclically shifted so that `split_edge(f) = f.e[0]`.

A edge is represented using an augmented half-edge data structure, where a pointer to the subdivided face (`fbase`) is included:

```
Hedge {
  Vertex*  point;
  Hedge*   mate;
  Face*    fbase;
}
```

These two data structures provide a compact way to encode the variable resolution 4–k mesh, as well as its inverse. They also make possible the efficient implementation of all relevant operations.

For example, the pre-image of a refinement $W(e)$ is

```
Set pre_image_w(Hedge e)
{
  return make_set(e.fbase, e.mate.fbase);
}
```

The image of a refinement $W(e)$ is

```
Set image_w(Hedge e)
{
  return make_set(e.fbase.children[0],
                  e.fbase.children[1],
             e.mate.fbase.children[0],
             e.mate.fbase.children[1]);
}
```

The successor refinement of a face f is

```
Hedge* successor_f(Face f)
{
  return split_edge(f);
}
```

The predecessor refinement of a face f is

```
Hedge* predecessor_f(Face f)
{
  return split_edge(f.parent);
}
```

The representation of type (ii) modification, corresponding to an edge swap, it uses the same data structures. The implementation is very similar. We take advantage of the fact the each face has only one child in the context of this operation. Thus, we set `f.children[1]=NULL`.

The effectiveness of a variable resolution structure can be analyzed according to the following criteria [8]: Expressive Power; Depth; and Growth Rate. The variable resolution 4–k mesh structure has all the desirable properties, as will be shown below.

A node of the DAG in the lattice representation of a 4-k mesh structure has some special characteristics because of the nature of the refinement operator W. The number of faces in the image $W_i(K_i)$ of W_i is always 4, and the number of faces in the pre-image of K_i of W_i is always 2. As a consequence, a node W_i has exactly two incoming arcs (the two nodes that generate the faces in the pre-image of W_i), and four outgoing arcs (the four nodes that reference one of the faces in the image of W_i). This is illustrated in Figure 5.

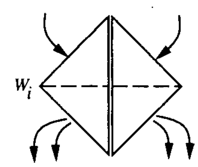


Figure 5: Node of the 4–8 DAG

In the following analysis we will consider the case of a semi-regular 4-8 variable resolution mesh, which is based only on edge splits. The 4–8 mesh features optimal properties among the family of 4–k meshes.

In a semi-regular 4–8 mesh, the initial mesh K_0 has arbitrary size, $|K_0| = n$. For a hierarchical structure with m levels, at each refinement step, $j = 1, m$, binary subdivision is applied to an independent set of two-face clusters that completely cover the mesh. Moreover, all clusters at subsequent levels j and $j + 1$ are interleaved. As a consequence, there are 2^j nodes in the DAG at each even level j. The size of the refined mesh produced by applying all modifications up to level j is $2^j\ n$.

The variable resolution structure of a regular 4–8 mesh has the following properties:

- *High expressive power*: It can be shown that the number, p, of distinct meshes produced by the 4–8 structure with m levels is equal to

$$p = \sum_{j=1}^{m} \sum_{k=0}^{2^j} \binom{2^j}{k}$$

  As an example, for $m = 6$, the expression power is $p = 18446744078004584724$.

- *Logarithmic depth*: the number of levels of a 4–8 structure with $q = 2^m$ nodes is approximately $l = \log_2 q$.

- *Linear Growth*: the growth rate is bounded by the ratio between the sizes of the image and pre-image of the modifications, which in the case of internal edge split is 2. It can be shown that the growth rate g of a 4–8 structure is bounded by $g = (n + 2)/(n + 1)$.

5 Construction Methods

This section gives an overview of the methods used to generate a variable resolution 4–k mesh.

We remark that it is important to have a variety of construction methods, so that they can be applied in distinct modeling situations, such as free form modeling, surface approximation, and conversion of representations.

The main categories of methods are the ones based on refinement and simplification.

Refinement-Based Methods: are subdivided into three types: semi-regular; quasi-regular; and irregular.

The *semi-regular* refinement method employs topology based subdivision. It generalizes the regular 4–8 mesh refinement and uses interleaved edge splits [11].

The method produces semi-regular meshes suitable for implementing stationary subdivision schemes. Figure 7 shows various subdivision surfaces generated with such schemes. The shape in this example is the "Stanford Bunny". The control polyhedron, shown in Figure 7(a), is a coarse mesh obtained from the original data through simplification [15]. The most natural scheme to implement using 4–8 semi-regular meshes is a generalization of subdivision for Box splines defined on four directional grids [11]. Figure 7(e) shows a C^1 subdivision surface based on the Zwart-Powell basis. Figure 7(f) shows a C^4 subdivision surface based on a degree 6 Box spline.

Because of the quadrangulated structure of semi-regular 4–8 meshes, it is also suitable for the implementation of subdivision schemes originally designed for quadrilateral meshes [3, 1]. This is achieved through a decomposition of primal and dual quadrilateral refinement into interleaved binary subdivision steps [14]. Figure 7(c) shows a biquadratic B-spline surface based on the Doo-Sabin scheme. Figure 7(d) shows a bicubic B-spline surface based on the Catmull-Clark scheme.

The *quasi-regular* refinement method employs geometry sensitive subdivision. At each level, it covers the mesh with two-face clusters selected using an edge length criteria. This method produces a mesh that combines quasi-regular 4–8 topology with almost uniform geometric features [17]. The quasi-regular mesh structure allows the implementation of quasi-stationary subdivision schemes. Figure 7(d) shows an example of a quasi 4–8 subdivision surface.

The *irregular* refinement method employs adaptive subdivision. It is based on multiresolution edge sampling. This method produces hierarchical meshes that conform to the shape of existing objects [13].

The irregular 4–8 mesh structure is suitable to adaptive surface tessellation. Because the subdivision algorithm is very general, it can work with both parametric or implicit surface descriptions. Figure 8 gives some examples of surfaces approximated by adapted irregular 4–8 meshes.

Figures 8(a) and (b) show a torus, defined implicitly. In Figure 8(a), we have an orthogonal projection of the base mesh together with the 3D grid; and Figure 8(b) the polygonal approximation. Figures 8(c) and (d), show the same torus, defined parametrically. In Figure 8(c) we have the adapted decomposition of the parameter domain; and in Figure 8(d) the polygonal approximation. The base mesh was simply the subdivision of the rectangle $[0, 2\pi] \times [0, 2\pi]$ along its diagonal into two triangles. The algorithm has structured the parameter domain into a 4–8 hierarchy with three layers.

Note that the algorithm produces consistent results using either the parametric or implicit description of a surface.

Figures 8(e) and (f), show a digitized bust of Spock. In Figure 8(f), we have an adaptive mesh which approximates the surface within a prescribed tolerance, and in Figure 8(e), we have the corresponding domain decomposition. The facial details are clearly visible, because the regions of high curvature are sampled more densely than the rest of the surface.

Simplification-Based Methods

Simplification-based methods construct the reverse of an increasing variable resolution 4-k mesh. They start with a fine mesh and coarsen it using the inverse of an edge split operation – an edge collapse. Therefore, they produce a decreasing hierarchical structure. For several reasons, it is advisable to establish the convention that the canonical lattice representation is an increasing structure, in which the source is a coarse mesh and the drain is a fine mesh. In this context, a simplification method builds the variable resolution representation "bottom-up".

In order to perform the simplification of a mesh with regular 4–8 connectivity, it is sufficient to apply the internal edge collapse operator that transforms a cluster of four faces into a cluster of two faces (see [15]). Moreover, the simplification procedure has to ensure that clusters at subsequent levels are interleaved. Unfortunately, this type of method is only practical for regular 4–8 meshes.

In the case of arbitrary meshes, it is necessary to use also the edge swap operator. The reason is that, since an irregular mesh does not have 4–8 connectivity, it may not be possible to cover the mesh with clusters of four faces sharing a degree 4 vertex $v \in V_4$. The edge swap operator is used to modify the mesh at each level, producing the required set of four-face clusters that cover most of the mesh [15].

Figure 9 shows an example of 4–k simplification. It is a cow model distributed with SGI's powerflip demo. The initial mesh, shown in Figure 9(a) contains 5800 triangles. The sequence of simplified meshes at levels 3 to 7, is shown in Figures 9(b) through (f). They contain respectively, 1200, 700, 400, 300, and 200 faces.

6 Level of Detail Operations

This section considers the application of variable resolution 4-k meshes for managing level of detail of large geometric models. It defines the relevant operations and gives some examples.

A level of detail operation consists in extracting a mesh K from a variable resolution structure $V = (K_0, W, \leq)$. As we have seen in section 3, this mesh $K \subset V$, corresponds to a front in the lattice representation of V, i.e., a set of arcs containing exactly one arc for each path from the source to the drain.

The collection of all nodes which can be reached from the source without traversing the arcs of the front, correspond to modifications to the mesh that are consistent with the partial order $\leq$ and produce the extracted mesh K. See Figure 6.

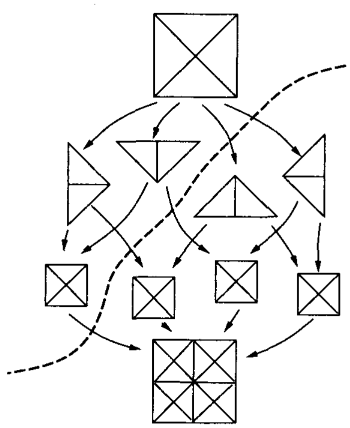


Figure 6: A front in the lattice representation.

We can abstract this level of detail operation as a *geometric query*, Q, to the variable resolution structure V.

This general query operation can be specified by the following parameters [4].

- An *adaptation function*: $\tau : K_j \to \{0, 1\}$, that computes some measure over V to determine if a face f produced by a modification should be accepted or not.
- A *focus set*: $S \subseteq \mathbb{R}^3$, that defines a region of interest where $\tau(f)$ is evaluated.

The answer to the query K, is the smallest conforming mesh such that $\tau(f) = 1$ and $f \cap F \neq \emptyset$, for all $f \in K_j$.

Examples of variable resolution query operations are: point location; region intersection; neighbor search and adapted mesh extraction. The last one is particularly important, because it appears in many graphics applications, such as, progressive rendering, real-time visualization and interactive modeling.

We remark that K could be either a mesh representing the whole surface, or a sub-mesh containing just the elements inside the region of interest. In the first case, the query is called *globally defined* and in the second case, *locally defined* [4].

Adaptive mesh extraction is implemented through a selective mesh refinement procedure using the variable resolution structure [4, 16].

This procedure can use a non-incremental or an incremental algorithm. The *non-incremental algorithm* is a specialization to 4-k meshes of the algorithms described in [8] for general variable resolution structures. It starts with an initial front that contains all the arcs leaving the source node, and gradually advances the front, in a top-down fashion, based on the evaluation of the adaptation function and intersection with the focus set.

The *incremental algorithm* uses an existing front, and updates it, moving the front up or down if necessary, according to the adaptation function.

We remark that the variable resolution structure guarantees that an extracted mesh is consistent *by construction*.

Another nice feature of this framework is that, the mesh extraction procedure is independent of the query specification. As a consequence, it is straightforward to incorporate it in completely different application domains. This gives a lot of flexibility from the systems design point of view.

In that context, what distinguishes two adapted mesh extraction operations is the nature of the adaptation function. Some common types of applications are related to: shape approximation; view dependent geometry, etc.

The practical performance of level of detail operations is highly influenced by the properties of the underlying structure, as noted by Puppo [8].

Next, we demonstrate the capabilities of the 4–k mesh structure in the context variable resolution queries.

Figure 10 exhibits few examples of adapted mesh extraction, using a variety of adaptation functions, as well as, variable resolution meshes constructed using different methods.

Figures 10(a) and (b) show two meshes representing a "saddle" surface that was defined parametrically. The variable resolution structure was constructed using adaptive refinement. The adaptation criteria used in Figure 10(a) was triangle size. In Figure 10(b) the criteria was intersection with a rectangular region in the parametric domain.

Figures 10(c) and (d) show two versions of the "Stanford Bunny". The one in Figure 10(c) was constructed using the C^4 Box-spline subdivision scheme; and the one in Figure 10(d) was constructed using simplification. The adaptation criteria is the same for both models: it is a linear ramp in the horizontal direction determining triangle size.

Figures 10(e) and (f) show an example of point location using the cow model of Figure 9. In Figure 10(e) we have the complete mesh, in which the smallest triangle was picked by pointing at the screen. A detail of the area surrounding this point is shown in Figure 10(d).

We close this section with some remarks about a useful capability of the 4–k mesh structure that allows the construction of a triangle strip representation of the extracted mesh [12]. Similarly to selective refinement, this algorithm starts with a path, defined on the coarsest mesh, and the path is refined while traversing the variable resolution structure. In particular, if the model has semi-regular 4–8 connectivity, it is possible to maintain a Hamiltonian path for all extracted meshes.

7 Conclusions

This section concludes the paper with a review of the results and a discussion of future work.

A framework for variable resolution description of surfaces was presented. It is based on the hierarchical 4-k mesh structure. This representation has several desirable properties for multiresolution applications.

We described various methods for constructing the 4–k representation that contemplate most modeling situations.

We also demonstrated the practical use of the 4–k representation, for the implementation of level of detail operations.

Future work in this area includes: hierarchical parametrizations; multiresolution decomposition; mesh compression; and the development of an integrated applications framework.

Acknowledgements

The figures in section 5 were generated with Geomview [7].

We acknowledge the Stanford Computer Graphics Laboratory and Viewpoint for providing the original data for the examples.

The authors are partially supported by research grants from the Brazilian Council for Scientific and Technological Development (CNPq) and Rio de Janeiro Research Foundation (FAPERJ).

References

[1] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Comput. Aided Design*, 10:350–365, 1978.

[2] Mark de Berg and Katrin Dobrindt. On levels of detail in terrains. In *Proc. 11th Annual ACM Symp. on Computational Geometry*, Vancouver, B.C., June 1995.

[3] D. Doo and M. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Comput. Aided Design*, 10:356–360, 1978.

[4] Leila De Floriani, Paola Magillo, and Enrico Puppo. Efficient implementation of multi-triangulations. *IEEE Visualization '98*, pages 43–50, October 1998. ISBN 0-8186-9176-X.

[5] Hugues Hoppe. Progressive meshes. In *SIGGRAPH '96 Proc.*, pages 99–108, Aug. 1996.

[6] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *SIGGRAPH '93 Proc.*, pages 19–26, Aug. 1993.

[7] S. Levy, T. Munzner, and M. Phillips. Geomview. Geometry Center, University of Minnesota, 1991.

[8] E. Puppo. Variable resolution triangulations. Technical report, CNR, 1998.

[9] E. Puppo and R. Scopigno. Simplification, LOD and multiresolution – principles and applications, 1997. Eurographics'97 Tutorial Notes.

[10] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, 1989.

[11] Luiz Velho. Semi-regular 4–8 refinement and box spline surfaces. Proceedings of SIBGRAPI, 2000.

[12] Luiz Velho, Luiz Henrique de Figueiredo, and Jonas Gomes. Hierarchical generalized triangle strips. *The Visual Computer*, 15(1):21–35, 1999.

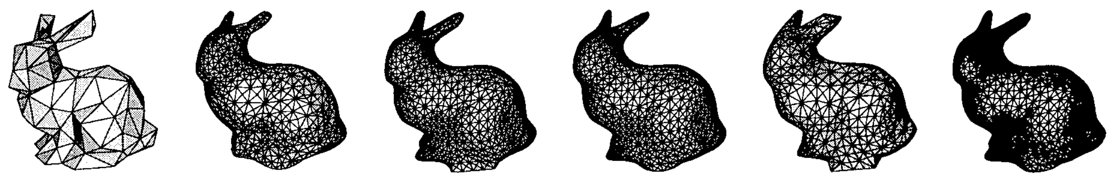
[13] Luiz Velho, Luiz Henrique de Figueiredo, and Jonas Gomes. A unified approach for hierarchical adaptive tesselation of surfaces. *ACM Transactions on Graphics*, 2000.

[14] Luiz Velho and Jonas Gomes. Decomposing quadrilateral subdivision rules into binary 4–8 refinement steps. preprint, 1999.

[15] Luiz Velho and Jonas Gomes. Four-face cluster simplification. preprint, 1999.

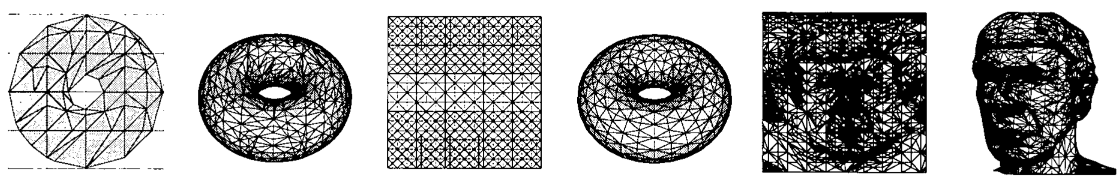
[16] Luiz Velho and Jonas Gomes. Level of detail operations using 4–k meshes. in preparation, 1999.

[17] Luiz Velho and Jonas Gomes. Quasi 4-8 subdivision surfaces. In *Proceedings of SIBGRAPI'99 - XII Brazilian Symposium on Computer Graphics and Image Processing*, Campinas, SP, Brazil, October 1999. SBC - Sociedade Brasileira de Computacao, IEEE Press.



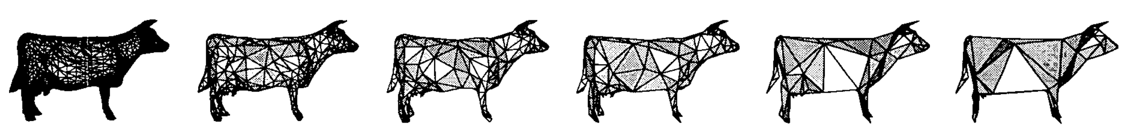
(a) Base Mesh (b) Quasi 4-8 (c) Doo-Sabin (d) Catmull-Clark (e) Zwart-Powell (f) C^4 spline

Figure 7: Surfaces generated by subdivision based on quasi 4–8 refinement (b), and semi-regular 4–8 refinement (c-f)



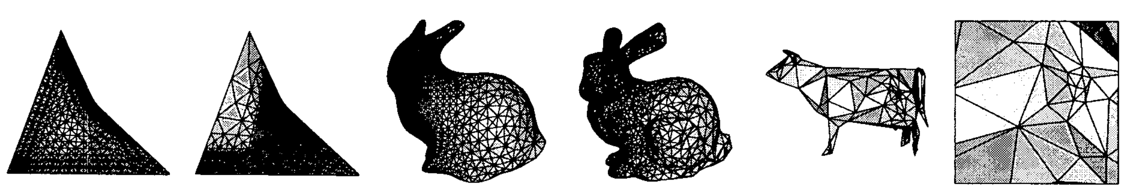
(a) (b) (c) (d) (e) (f)

Figure 8: Approximations using adaptive 4–8 refinement of implicit (a-b), parametric (c-d), and sampled (e-f) models.



(a) 5800 faces (b) 1200 faces (c) 700 faces (d) 400 faces (e) 300 faces (f) 200 faces

Figure 9: 4-k Simplification of a cow model.



(a) (b) (c) (d) (e) (f) detail of (e)

Figure 10: Adapted variable resolution 4-k mesh extraction.